CDE2310: Fundamentals to System Engineering

AY 2024/2025 Semester 2

Group 4

Names:	Matric Number:
Toh Leong Chuan	A0272820J
Charlie Hee Ming Guan	A0271614J
Manya Gupta	A0309673W
Muhammad Irfan Bin Abdullah	A0273164E

Table of contents

Chapter 1: Introduction

1.1 Problem Definition

The objective of this project is to design and construct an autonomous robotic system, specifically utilising the Turtlebot, to navigate an unfamiliar maze environment in order to detect heat signals that may indicate the presence of survivors. Once potential survivor locations are identified, the robot will signal search and rescue teams by deploying flares (represented by ping pong balls) at each detected heat signal zone. This operation must be completed without relying on line-following navigation techniques and must adhere to a strict timeframe of 27.5 minutes for both setup and mission execution. Additionally, there is a challenge that involves identifying and ascending a ramp using a 2-D lidar sensor to locate a third survivor zone, which offers bonus points but is not critical to the primary mission.

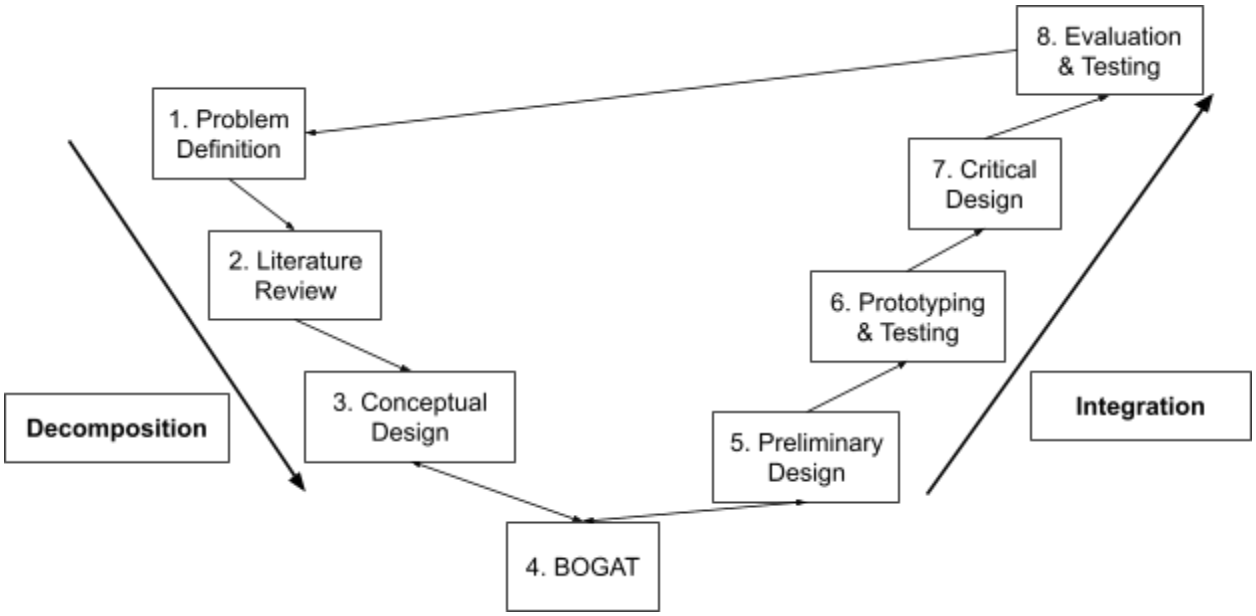


Figure 1.1a: V-Model used in CDE2310

Mission #1 Map

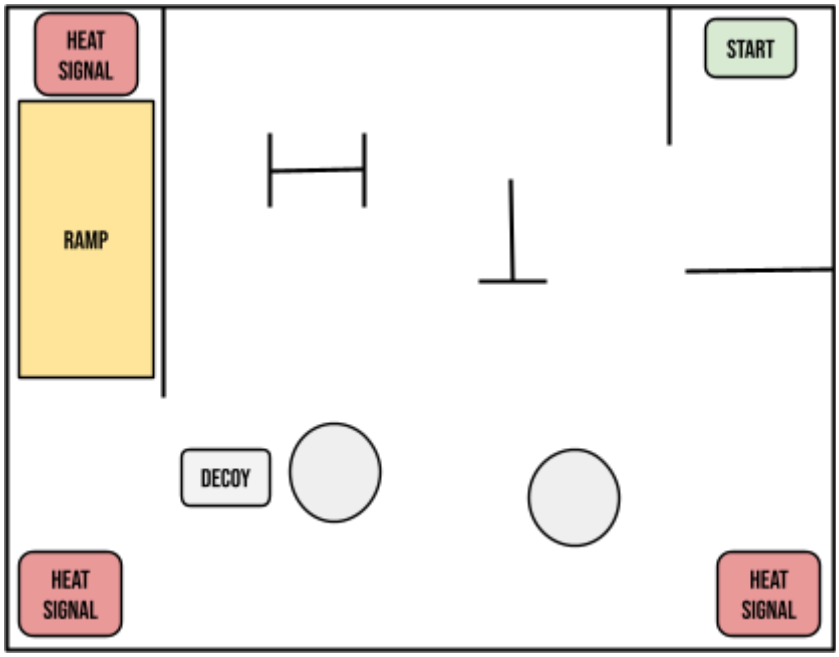


Figure 1.1b: Mission Map

1.2 System Requirements

1.2.1 Project Deliverables

For this mission, the project deliverables are defined to simulate a search and rescue scenario. A set of functional and performance-based requirements was established at the outset of the mission.

No.	Project Deliverables
1	System shall incorporate a payload integrated with navigation, heat sensing, and a launching mechanism
2	System shall autonomously navigate an unknown closed environment while avoiding obstacles
3	System shall develop ROS 2 Humble-based software stack for autonomous navigation, heat detection and ping pong ball deployment
4	System shall detect and accurately identify the heat signature
5	System documentation, including setup procedures, architecture diagrams, and usage guidelines

Table 1.2.1: Project Deliverables

1.2.2 Functional Requirements

No.	Functional Requirements
1	System shall autonomously navigate an unknown maze environment without collision
2	System shall detect 2 heat signatures and approach the heat signal zone using onboard sensors
3	System shall fire 3 ping pong balls at intervals of 2-4-2 seconds at every heat signal zone
4	System shall be capable of carrying and deploying 9 ping pong balls

Table 1.2.2: Functional Requirements

1.2.3 Non-functional Requirements

Requirement	Performance Requirements
Performance	System shall complete the detection and signalling using ping pong balls within 27.5 minutes, including setting up the system
Reliability	System shall accurately detect new heat signals within a 0.5 metre range
Usability	System shall be designed such that it is easy to set up and operate
Accuracy	System shall deploy every ping pong ball at a minimum height of 1.5 metres
Verification	System shall record the exploration map by generating a SLAM map
Safety	System shall prioritise safety and avoid wall collision

Table 1.2.3: Performance Requirements

1.2.4 Constraints

Constraint	Description
Physical Size	The robot must fit within size restrictions and cannot exceed the height of the maze walls. Furthermore, a wider robot complicates navigation through narrow passages.
Power Supply Limitation	The robot must operate within the constraints of the given power source.
Sensor and Actuator Limitations	The selection of electronic components, including the sensors and the actuators, must factor in cost, compatibility and power output
Interface	The hardware components must be interoperable with the ROS2 and Turtlebot ecosystem
Environment	The robot will operate in an enclosed environment. Environmental factors like the lighting, terrain and heat noise might affect the overall performance of the robot
Firing Angle	The ping pong balls launched by the robot must not come into contact with any walls or obstacles during their upwards trajectory.
Cost	The fabrication and development of the robot must be limited to a budget of $80. Most resources should be procured without monetary means.

Table 1.2.4: Mission Contraints

1.3 Safety Requirements

Safety considerations are integral when designing and handling the robot. Appropriate safety mechanisms and precautions shall be incorporated to prevent harm or malfunction during operations

1.3.1 Safety Design Considerations

The robot is most dangerous when the firing mechanism is activated. The robot's firing mechanism will be meticulously designed to prevent any accidental discharge of ping pong balls, ensuring the system operates safely and effectively. Additionally, safety measures will be implemented to guarantee that the fired ping pong balls pose no risk of injury. When the robot is firing, avoid touching any part of the robot, especially the flywheels.

1.3.2 Handling

When handling the system, it is essential to exercise extreme care and attention to detail. Avoid exposing the system to harsh conditions, such as dropping, burning, puncturing, crushing, or water exposure, as these actions can jeopardise its integrity and functionality. If any damage is observed, immediately disconnect the power supply and conduct a thorough inspection for repairs.

Chapter 2: System Overview

2.1 Integrated System

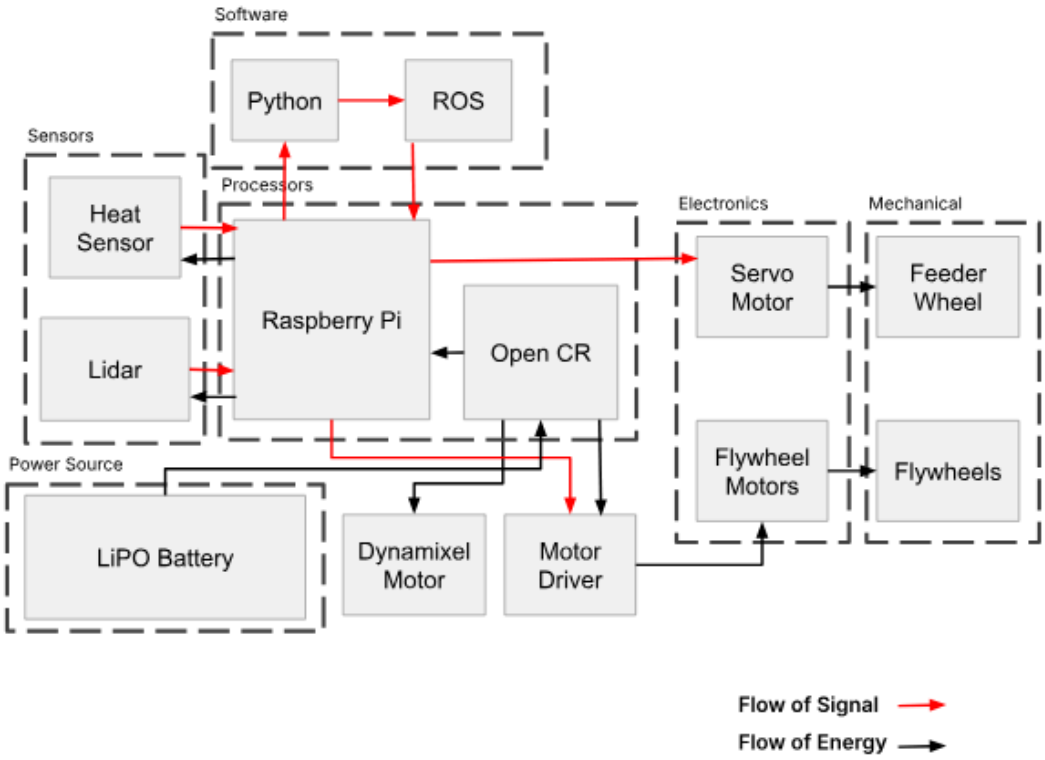


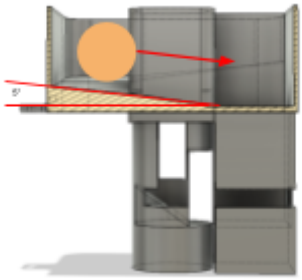
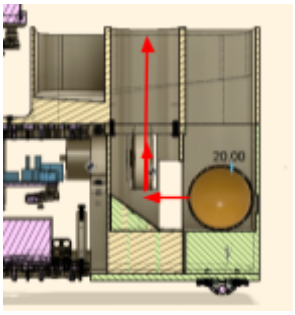
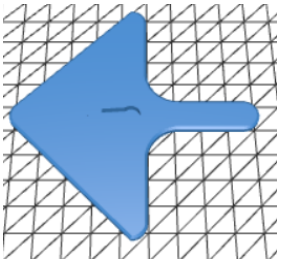
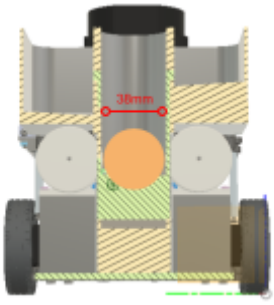
Figure 2.1 Integrated System

2.2 Concept of Operations

The robot is designed to autonomously navigate an unknown maze-like environment to locate simulated survivors represented by heat signals. Upon detection, it signals the survivor's location by firing ping pong balls (flares). The desired outcome is for the robot to locate two heat signals at 2 different locations and launch 3 ping pong balls at each location. The robot is also designed to scale a ramp to reach an additional survivor zone for bonus objectives.

2.2.1 Firing Mechanism Concept

The Firing Mechanism is centred on the robot's payload, which includes a hopper unit designed to hold ping pong balls and a firing system consisting of a pair of flywheels to launch the balls. Additionally, a servo motor controls a ball feeder that delivers the balls one at a time.

Concept	Function
Hopper Slope 	The sloped base of the Hopper enables passive ball feeding. As each ping pong ball is launched, gravity naturally causes the remaining balls to roll down toward the launching mechanism without requiring additional actuators. At 5°, this slope ensures a continuous and reliable supply of balls for firing at the required intervals.
Ball Feeder Mechanism  **Ball Feeder** 	The ball feeder is a servo-driven mechanism designed to deliver one ping pong ball at a time into the flywheel launcher. When ready to fire, the servo rotates the feeder arm, pushing a single ball onto a slope positioned beneath the flywheels. This slope guides the ball upward into the flywheel contact zone, where it is propelled forward and launched. The geometry of the feeder arm ensures that only one ball is released at a time, partially blocking the next ball from falling until the arm returns to its original position. This controlled motion prevents continuous feeding. To maintain a clear and timed signal, the servo is programmed with specific delays: a 4-second wait before feeding the second ball and a 2-second wait before the third.
Flywheels 	The flywheel mechanism is responsible for launching the balls once they are fed into position. Two flywheels are mounted in parallel with a 38 mm gap between them, slightly less than the 40 mm diameter of a standard ping pong ball. This 2 mm compression ensures firm contact, generating the necessary friction to accelerate the ball effectively through the launcher. To enhance grip and reduce slippage, the surface of each flywheel is coated with a layer of silicone glue, which provides additional traction during ball launch. The flywheels are powered by DC motors and are kept continuously spinning throughout the firing sequence. This design allows for immediate ball launch upon contact.

2.2.2 Heat Signature Detection

The robot uses the AMG8833 infrared thermal camera to detect heat signatures. The AMG8833 captures temperature data in an 8×8 grid, providing real-time thermal imaging of the robot's surroundings. Each pixel in the grid corresponds to a temperature reading, allowing the system to localise heat sources with reasonable spatial resolution.

The thermal data is continuously transmitted to the Raspberry Pi via an I²C communication interface. The onboard software processes this data to identify any grid cells exceeding a predefined temperature threshold. When a temperature of 27°C or higher is detected in any cell, it is interpreted as a potential survivor signal.

Upon confirming a heat source, the robot initiates a target-approach behaviour, adjusting its path to centre the heat signal in its field of view. This enables it to align accurately with the source before launching a signalling flare, ensuring the robot only responds to meaningful thermal cues while ignoring environmental noise or false positives.

2.3 System, Technical Specifications

2.3.1 Turtlebot3 Specifications

Components	Specifications
Maximum translational velocity	0.21 m/s
Maximum rotational velocity	2.84 rad/s (162.72 deg/s)
Size (L x W x H)	138mm x 178mm x 192mm
Weight	1kg
Expected operating time	2h 30m
Sensors	AMG8833, LiDAR 2.0
Actuators	Servo motor, L298N Motor Driver, 12V RS-3835 Motor
Power connectors	3.3V / 800mA 5V/4A 12V/1A
Power adapter (SMPS)	Input: 100- 240v, AC50/60Hz, 1.5A @max Output: 12VDC, 5A

Table 2.3.1: Specifications of Turtlebot3

2.3.2 Payload Specifications

Component	Specifications
AMG8833 Heat Imager	● Sensor Array: 8x8 (64 pixels) ● Temperature Range: -20°C to 100°C ● Frame Rate: 1 to 10 Hz ● Interface: I2C ● Supply Voltage: 3.3V to 5V ● Operating Current: 4.5mA (typical) ● Viewing Angle: 60 degrees ● Mounting Form: Surface Mount Type
Flywheel DC Motor RS-385 12V DC Motor	● Operating Voltage: 6V to 14V DC ● Rated Voltage: 12.0V ● No-load Speed: 10000 ±1500 RPM ● No-load Current: 70 mA max ● Loaded Speed: 4500 ±1500 RPM ● Loaded Current: 250 mA max ● Stall Current: 500 mA max ● Starting Torque: 20 g*cm ● Rated Load: 10 g*cm ● Operating Temperature: -10°C to +60°C ● Body Size: 27.5mm x 20mm x 15mm ● Shaft Size: 8mm x 2mm diameter ● Weight: 17.5 grams
L298N Motor Driver	● Operating Voltage: 5V ● Motor Supply Voltage: 5V to 35V ● Logic Input Voltage: 2.3V to 7V ● Max Output Current: 2A continuous ● Control Inputs: IN1, IN2, IN3, IN4 ● Enable Pins: ENA, ENB (PWM controlling speed) ● Dimensions: 43 mm × 43 mm × 27 mm ● Cooling: On-board heatsink
Servo Motor	● Operating Voltage: 4.8V to 6.0V ● Operating Current: ~100 mA (idle) to 650 mA (stall) ● Stall Torque: 1.8 kgcm at 4.8V ● Speed: 0.12 sec/60° at 4.8V ● Rotating Range: 0° to 180° ● Control Signal: PWM ● PWM Frequency: 50Hz (20ms period) ● Weight: 9 grams ● Dimensions: 22.8 mm × 12.2 mm × 28.5 mm

Table 2.3.2a: Specifications of Electrical Components

Feature	Specifications
3D Printed Material	PTEG: • Hopper • Launcher base • AMG8833 Holder • Modified Waffle Plate for LDS • Modified Waffle Plate for L298N • L298N Holder TPU: • Ball Feeder • Flywheel • Motor Holder • Servo Adapter
Weight of Payload	434 grams
Total Weight	1394 grams
Dimensions	213 mm x 178 mm x 192 mm

Table 2.3.2b: Specifications of Robot

2.4 Mechanical Subsystem Design

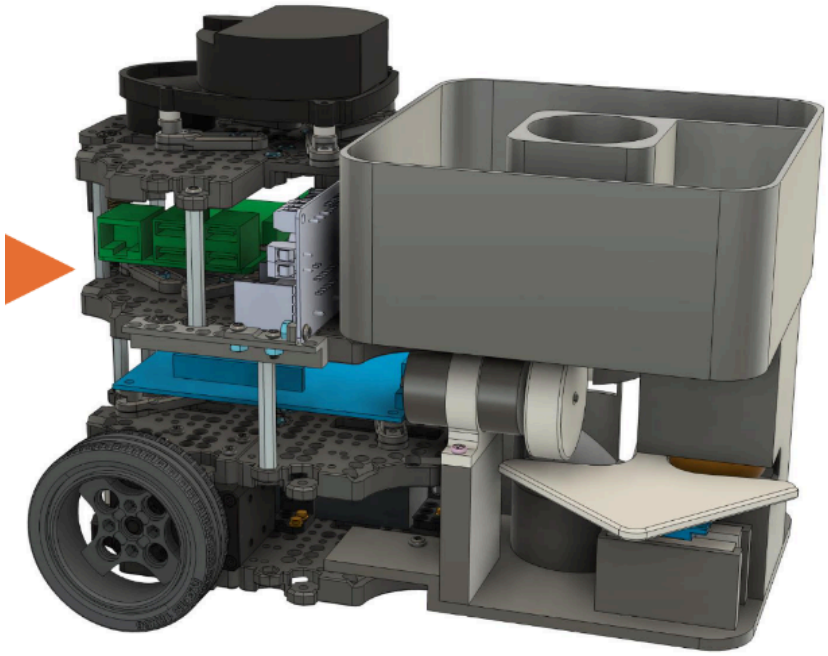


Figure 2.4: Final Iteration

Table 2.4 presents the calculations for the center of mass for the robot, excluding fasteners such as screws, nuts, and rivets, as their mass was too negligible to record.

Parts	Mass (g)	X CG from origin (cm)	Y CG from origin (cm)	Z CG from origin (cm)	M x Xcg	M x Ycg	M x Zcg
Waffle Plate Lv1	80	6.75	9	1.3	540	720	104
Waffle Plate Lv2-4	240	6.75	9	9.8	1620	2160	2352
Ball Caster	4	12.5	9	0.3	50	36	1.2
Left Motor	56	4.75	14.25	3.1	266	798	173.6
Right Motor	56	4.75	3.25	3.9	266	182	218.4
Left Wheel	27	3.25	16.6	3.25	87.75	448.2	87.75
Right Wheel	27	3.25	1.4	3.25	87.75	37.8	87.75
Li-Po Battery	141	7.7	9	3.05	1085.7	1269	430.05
Level 1 supports(M3x35)	20	6.75	9	3.6	135	180	72
Level 2 supports(M3x45)	24	4.65	9	8.2	111.6	216	196.8
Level 3 supports(M3x45)	36	8	9	13	288	324	468
OpenCrR1.0 + supports + cable	69	6.45	9.3	7.5	445.05	641.7	517.5
Raspberry Pi + supports + cable	50	10.5	5	11.5	525	250	575
LDS-02	128	3.9	9	17.4	499.2	1152	2227.2
USB2LDS	2	3.3	9	11	6.6	18	22
Payload							
Motor Driver	33	8.1	14.2	15.7	267.3	468.6	518.1
Motors + flywheel + motor mount	170	12.8	9	8.1	2176	1530	1377
Storage unit + firing barrel	167	16.6	9.4	8.9	2772.2	1569.8	1486.3
Servo motor + feeder	60	18.3	13.9	3.8	1098	834	228
AMG8833 + AMG8833 mount	4	-1.4	9	12.7	-5.6	36	50.8
total mass	1394						

Table 2.4: Centre of Mass

The centre of mass was determined to assess the robot's balance. Ideally, the centre of mass should be situated just below the midpoint of its height and slightly forward of the robot's centre. The computed coordinates for the centre of mass are x = 8.84 cm, y = 9.23 cm, and z = 8.03 cm.

2.5 Electrical Subsystem Design

2.5.1 Electronics System Architecture

The Electronics System Architecture illustrates the flow of data and energy between electrical components.

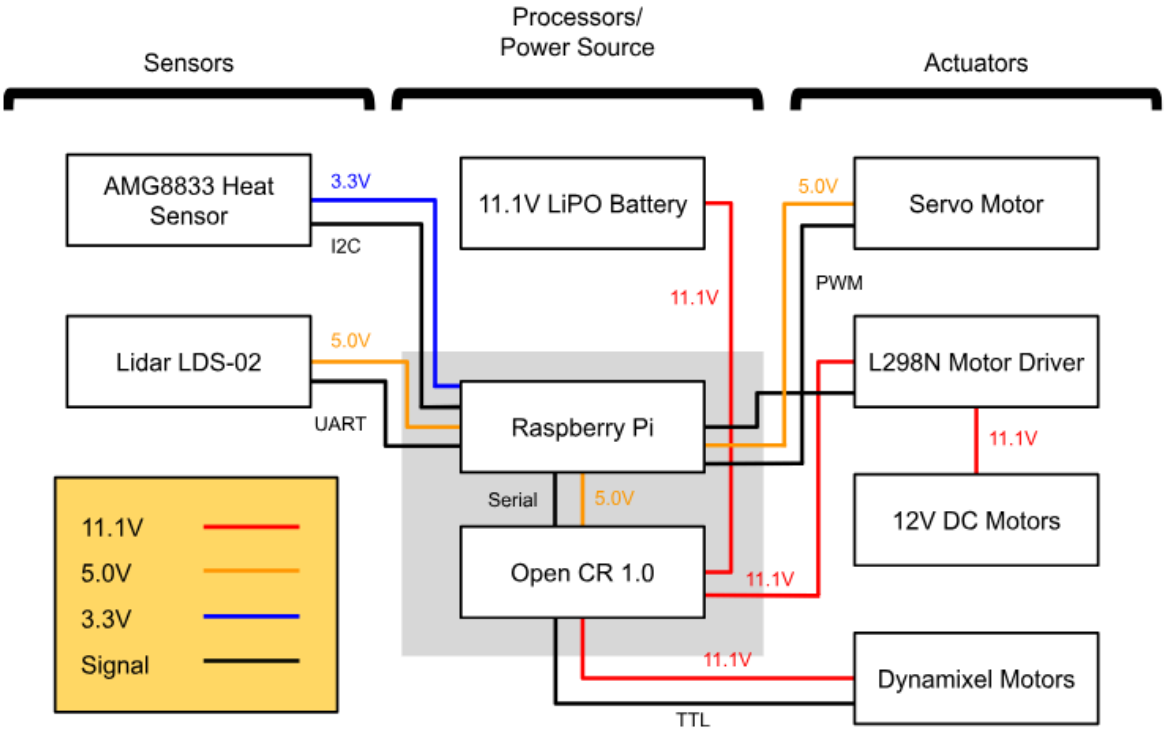


Figure 2.5.1: Electronics System Architecture

2.5.2 Power Consumption

Components	Voltage (V)	Current (A)	Qty	Power (W)	Remarks
During Boot Up					
Initial Boot Up	11.09	0.634	1	7.03	As the buzzer will sound
				7.03	

Standby					
During Standby/Idle	11.09	0.523	1	5.86	-
L298 Motor Driver*	2	2	1	4	66% duty cycle
Servo Motor	4.99	0.153	1	0.763	Maximum duty cycle
12V DC Motor*	11.1	0.26	2	5.88	
				16.5	
Operations					
During Operation, rteleop	11.09	0.733	1	8.14	Measured during maximum translational velocity = 0.22 m/s
AMG8833 Infrared Array Sensor*	3.3	0.0045	1	0.01485	Operating at 10 fps
				8.155	

**Taken from the respective datasheet*

Power Budget

Calculations were conducted to determine the total number of trials possible with a fully charged battery. Since different operations are done separately, for example, when the robot is navigating, components involved in launching the ping pong balls are not in use and thus do not draw power, the calculations were done based on the estimations on how long each operation will last.

Time for Boot up = 2s
Time for Standby = 50s
Time for Operation = 300s

Power for Boot up = 7.03 W
Power for Standby = 16.5 W
Power for Operation = 8.155 W

Power needed = (7.03*2 + 16.5*50 + 8.155*300)/352 = 9.33 Watts
Assuming every component is running at the same time, the LiPo battery can last:

11.09V * 1800mAh / 9.33 = **2.139 h = <u>128.37 mins</u>**

Given our total running time is roughly 6 minutes, we can run approximately 21 cycles.

2.5.3 Wiring Convention

The Wiring Convention specifies the connection points and the gauge of wires used throughout the system.

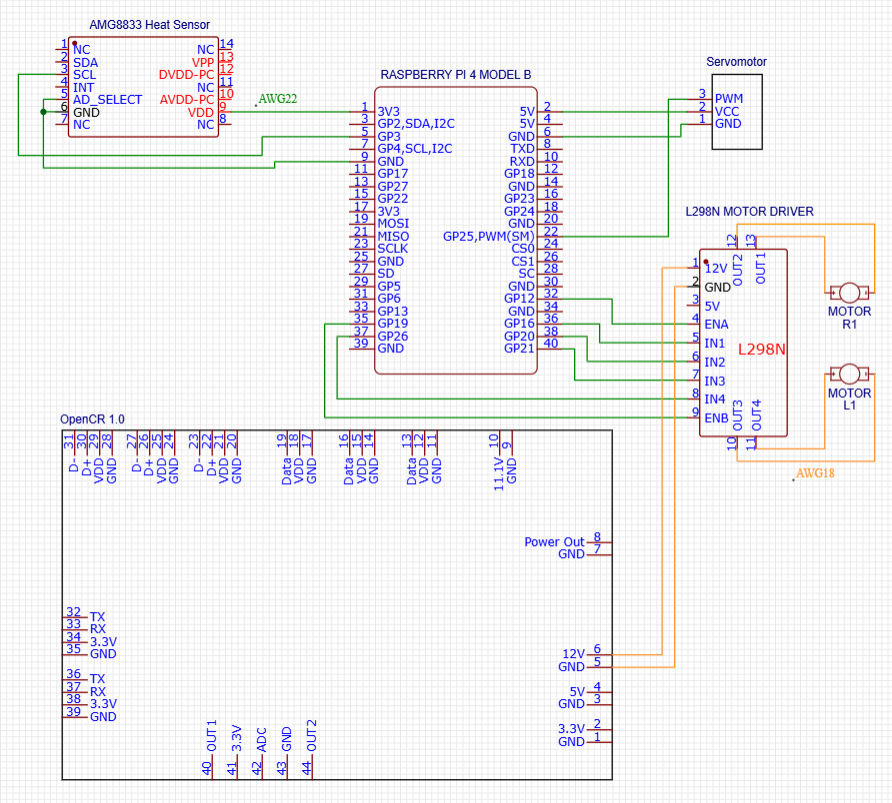


Figure 2.5.3: Wiring Convention

In the schematics, the GPIO pins are referenced according to the Raspberry Pi 4 configuration. The infrared sensor labelled AMG8833 has four connections to the Raspberry Pi: Vin is connected to 3.3V, GND to GND, and SCL and SDA to GPIO pins 2 and 3, respectively. The servo motor is connected through GND, 5V, and GPIO 18 for the GND, 5V, and PWM wires. Additionally, GPIO pins 12, 16, 20, 21, 26, and 19 are connected to ENA, IN1, IN2, IN3, IN4, and ENB, respectively. There is also a connection to the GND of the Raspberry Pi.

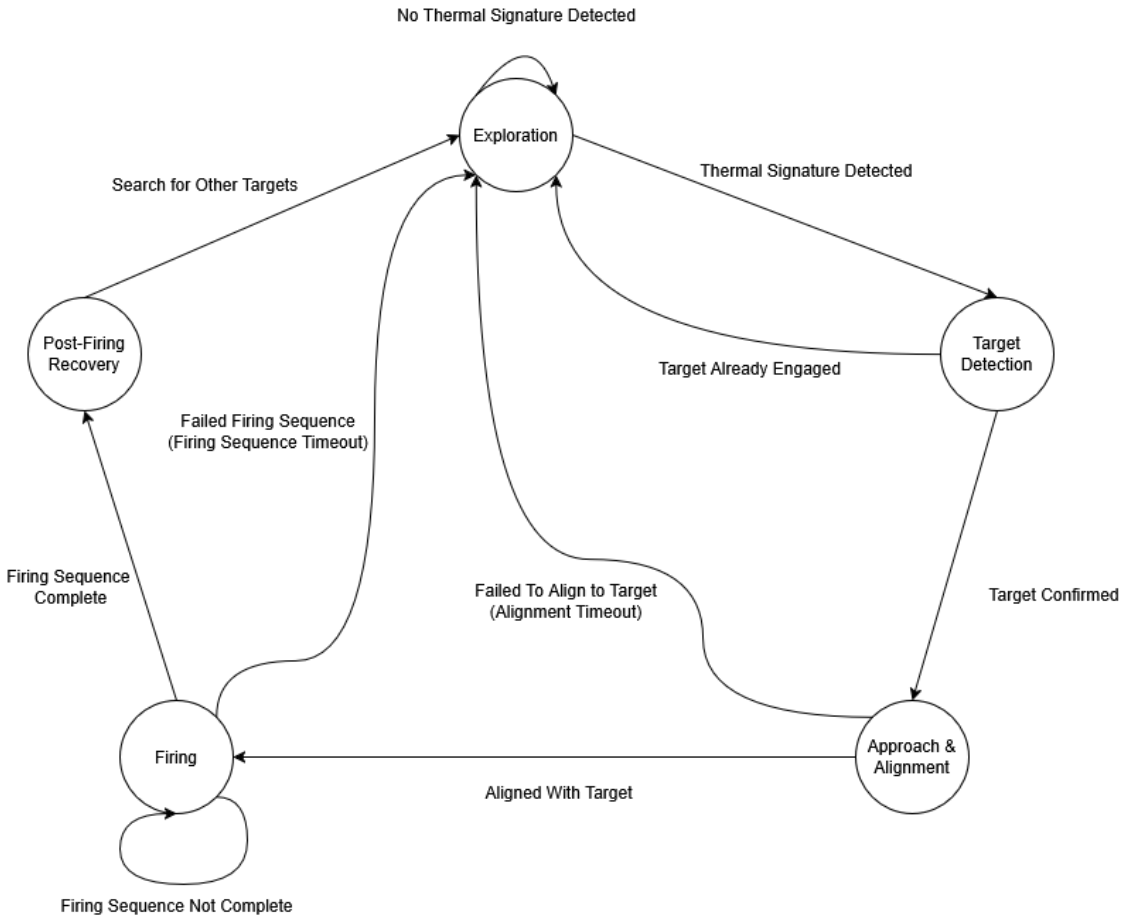
AWG No.	Max Current [Amperes]	Resistance [Ohms/km]
18	2.3	20.94
24	0.557	84.20

Table 2.5.3: Wire Gauge Specifications

Connections from the OpenCR module to the L298N motor driver, as well as from the L298N motor driver to the 12V DC motors, due to their suitability for higher voltage and current

applications. AWG 24 wires are used for the remaining connections, offering adequate performance while being easier to handle and more suitable for iterative testing.

2.6 Software Subsystem Design



The software is structured around ROS2's modular node-based architecture, where each node is designed to perform a single, clearly defined function. This modularity supports scalable development, ease of debugging, and efficient communication between subsystems.

A Finite State Machine (FSM) serves as the core control logic, managing transitions between mission-critical tasks. The main states in the FSM include:

- **Exploration**: Autonomous maze traversal using lidar and odometry.
- **Target Detection**: IR signature scanning to identify valid targets.
- **Alignment**: Positional adjustments to face the target.

- **Firing**: Execution of the firing sequence via servo and solenoid.
- **Evaluate/Repeat**: Assessment of task completion and transition logic.

This FSM-based, node-centric design enables separation of concerns and lays the foundation for simulation and hardware integration, ensuring consistency across development and deployment phases.

Chapter 3: System Fabrication Procedures

This section provides a guide to the robot fabrication process, which is divided into three parts: mechanical, electrical, and software setup.

3.1 Mechanical Fabrication and Assembly

The robot builds upon the standard Turtlebot3 Burger. Other than fasteners and electrical components, all additional parts are 3D printed. This includes mounts for L298N motor driver, AMG8833, two flywheel motors and a positional servo. To ensure a rigid assembly, all components are solidly connected using nuts and screws with the aid of some heat set inserts. The end product is mechanically sturdy and robust.

Instructions on how the robot can be fabricated and assembled can be found in the following link:
https://github.com/t-leongchuan/r2auto_nav_CDE2310_grp4/blob/main/docs/Mechanical%20Instructions.pdf

3.2 Electrical Assembly

To simplify troubleshooting and avoid the hassle of tracing wires to the correct pins, all wires from an electrical component are labelled with electrical tape for easier identification. Additionally, the lengths of the wires were measured and shortened to prevent disorganised and cluttered wiring.

3.2.1 L298N Motor driver

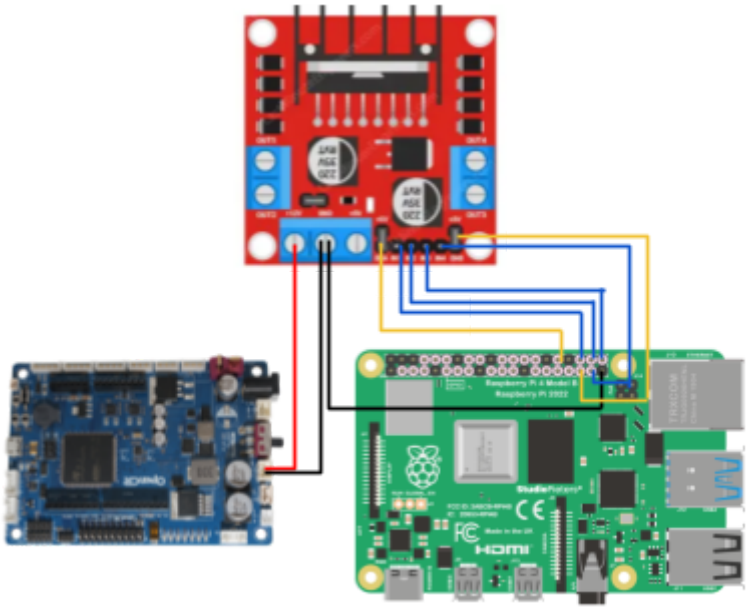


Figure 3.2.1: L298N Motor Driver Connections to OpenCR and Raspberry Pi

The wires connecting to the OpenCR are screwed in 12V and GND pins of the L298N. The rest of the wires that connect the L298N to the Raspberry Pi are done using jumper wires.

3.2.2 Motor

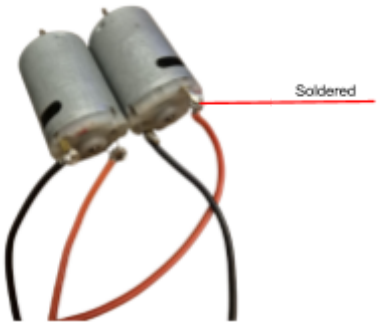


Figure 3.2.2: Soldering of 18 AWG Wires to RS385 DC Motor

To ensure a secure and permanent connection, 18 AWG wires were soldered directly onto the DC motor, as shown in Figure 3.2.2. The direction of rotation of the DC motor can be altered by code, thus, there is no need for colour coding of the wires.

Chapter 4: Bill of Materials

No.	Component	Description	Qty	Price	Procurement Method
Mechanical					
1	Launcher Base	3D printed parts ● Launcher Base and Hopper were printed twice Total Duration paid for: 8 hrs, $5 per hour	1	$40	3D Print
2	Hopper		1		
3	AMG8833 mount		1		
4	Motor mount		2		
5	Flywheels		2		
7	Rotating feeder		1		
8	Servo Motor Adapter		1		
9	PH_M3x8mm_K	-	10	-	E2A Lab
10	PH_M2.5x8mm_K	-	8	-	
11	PH_M3x6mm_K	-	12	-	
12	NUT_M2.5	-	8	-	
13	NUT_M3	-	22	-	
Electronic					
1	AMG8833	Heat sensor	1	-	E2A Lab
2	Positional Servo Motor - SG90	Servo Motor	1	-	E2A Lab
3	DC Motor - RS-385	Flywheel Motor $5.34 each	2	$10.64	Purchased
4	L298N	Motor Driver	1	-	E2A Lab
Total			5	$50.64	

Table 4: Bill of Materials

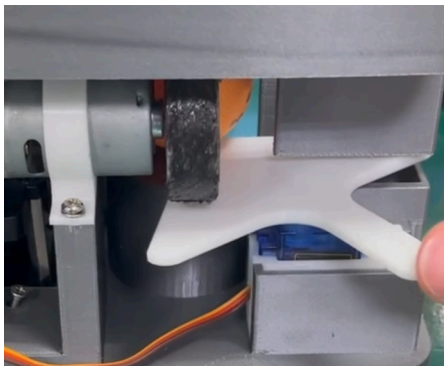
Chapter 5: Prototyping and Testing

5.1 Mechanical Prototyping & Testing Process

We decided early on that we wanted a flywheel mechanism because we felt it minimised system complexity. Although we did not know how to implement our idea fully, we got to work and figured out the design as we progressed. Details of how we went from our first idea to our final design can be found in the PDF below.
https://github.com/t-leongchuan/r2auto_nav_CDE2310_grp4/blob/main/docs/Mechanical%20Prototyping%20%26%20Testing.pdf

5.2 Electrical Prototyping

The feeder is initially operated by a 360° continuous servo motor MG995. The objective was to ensure that the ping pong balls are fed to the flywheel one at a time by controlling both the speed and direction of the feeder. However, it became clear that the MG995 model was unsuitable for the design due to several issues.

Initial Design	Final Design
The height of MG995 obstructs clearance between the feeder arm and the flywheel, preventing the ball from being launched at the desired height.  An additional challenge encountered was the difficulty in achieving precise positional control with a continuous servo motor, leading to instances where the motor overshot its target and unintentionally fed multiple balls at once, typically resulting in the simultaneous firing of at least two balls. To address this issue, the design was revised to incorporate a positional servo motor, which provides accurate angular control.	 We have replaced the MG995 continuous servo motor with an SG90 positional servo motor. An adapter was 3D printed to accommodate the smaller size of the SG90 for assembly. While the positional servo motor does not offer the full range of motion, filling the arms of the feeder allows it to be reset, enabling it to feed the balls again.

5.3 Preliminary Software Design

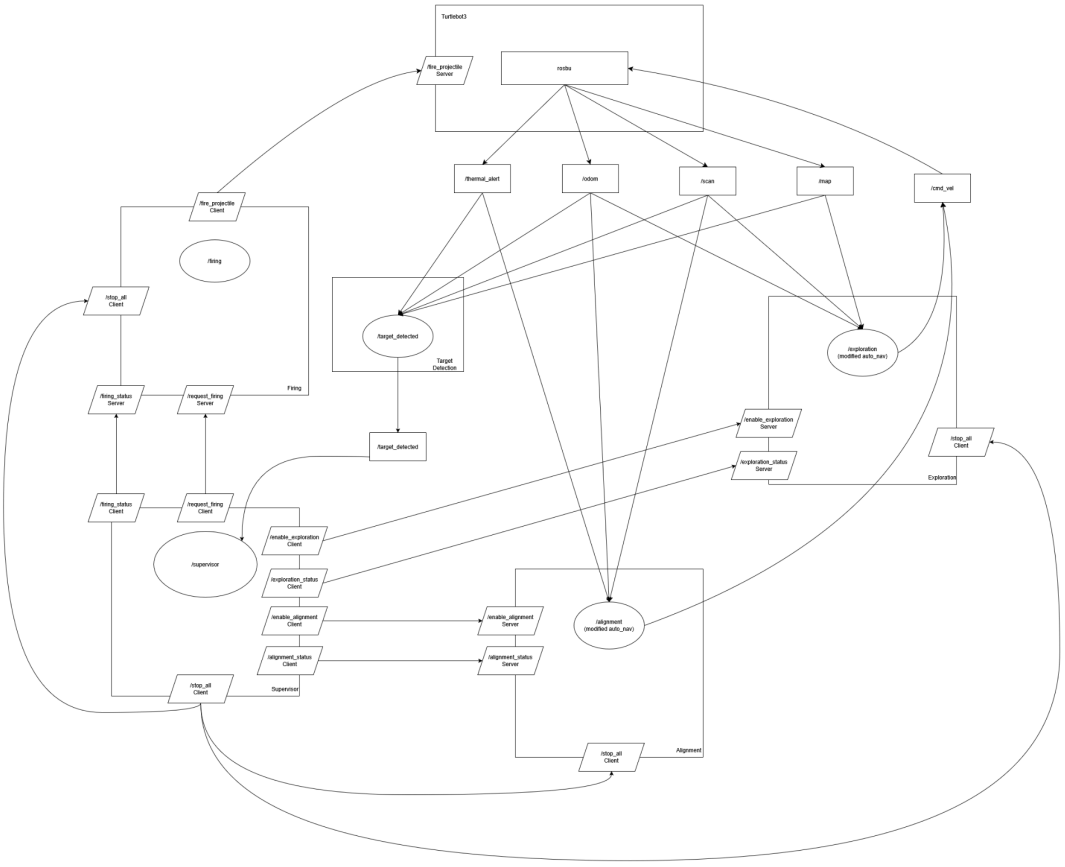


Figure 5.2.1: ROS2 Node Graph (RQT)

Refer to the following link for the RQT Graph https://imgur.com/a/BD1oLtm

Building upon the concept design's use of a Finite State Machine (FSM), the ROS2 node architecture is structured to reflect these states through an organised RQT graph. The system emphasises modularity through the use of ROS2's client-server model for state transitions.

At the centre of this design is a **supervisor node**, which functions solely as a state manager. It does not directly process sensor data or issue actuation commands. Instead, it coordinates transitions between states by initiating service calls to dedicated nodes responsible for specific tasks, such as navigation, target detection, alignment, and firing.

This design enforces a clear separation of concerns. Each node operates independently and only engages when requested by the supervisor, allowing for individual testing, replacement, or extension without impacting the rest of the system. Such modularity also supports smoother integration and easier debugging during the development and testing phases.

5.3.1 Trial Test

Autonomous navigation was primarily tested using the Gazebo Turtlebot3 simulation, allowing for convenient and repeatable testing without requiring a physical maze setup.

Basic unit tests were conducted on individual software components. Navigation was tested independently, and thermal sensor performance was verified by running the sensor node and using teleoperation (via `rteleop`) to simulate environmental conditions, observing real-time sensor feedback and identifying an appropriate temperature threshold.

Each electrical component was also tested individually before executing a full sequence that included the firing mechanism. This allowed us to evaluate the interaction between electrical and mechanical subsystems.

While some unit testing for subsystems like Navigation, Alignment, and Firing was conducted, it was not extensive enough to surface all potential integration issues. Constraints in time and resources limited the depth and coverage of these tests.

Chapter 6: Operational Plan

6.1 Mission Plan

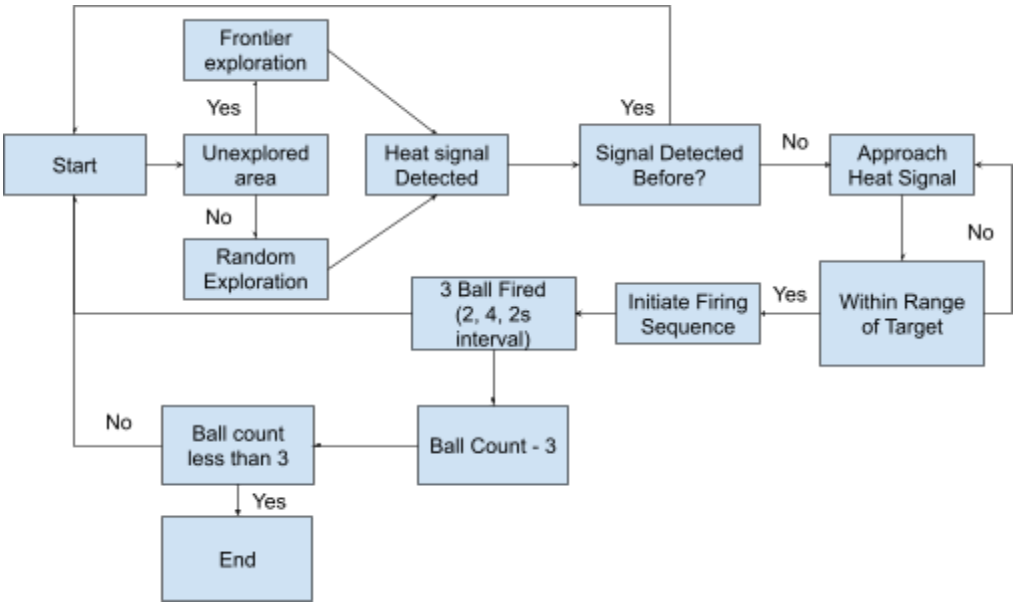


Figure 6.1: Mission Plan

6.2 Pre-Op Check

Component	Intended Condition	Pre-operations Check	Intended Observation
OpenCR	Able to be powered by the LiPO Battery	Turn the OpenCR on	Green LED lights up and the boot-up tune is played
Raspberry Pi	Able to connect to network and be accessible from a remote laptop	Run “sshrp” from a remote laptop connected to the same Wi-Fi network	Terminal hostname change to “ubuntu” when “sshrp” is run on laptop
LiDAR	Able to spin and collect data	Run “rosbu” on the Pi, and “rslam” on the laptop	Mapping with SLAM works, and it is displayed in RViz
AMG8833	Able to detect heat signature	Place a heated item, like a palm, in front of the sensor	There is a difference in temperature reading when the palm is in front of the sensor
L298N Motor Driver	Able to be powered by the OpenCR	Turn the OpenCR on	Red LED lights up on the motor driver
Servo Motor	Able to rotate and stop at an accurate position	Run the firing code	The arm of the feeder attached turns to feed the ball to the flywheel and returns to its original position
Flywheel	The silicone layer on the flywheel is not peeling off	Turn and touch every part of the flywheel	There should not be an exposed TPU surface that is going to be in contact with the ball launched
Structural Stability	Structural platforms and components are installed correctly	Shake robot	All components are mounted securely, and no loose parts are heard or fall off.

Table 6.2: Pre-Op Checklist

Chapter 7: Testing and Validation

7.1 Maze exploration and Navigation

During the final run, the robot initially failed to move from its starting position. After reviewing the sensor thresholds, the team decided to lower the detection threshold from 8 to 6, which enabled the robot to begin its movement. However, it immediately collided with the wall upon departure. This pattern of movement followed by collision continued through several attempts, ultimately resulting in unsuccessful autonomous navigation. At the 19-minute mark, the team made a strategic decision to manually intervene, guiding the robot toward the heat signal zone using teleoperation, prioritising partial task completion over the ongoing autonomous failures.

7.2 Navigation Failure – White Flag

7.2.1 Firing Sequence

After White Flag, the firing mechanism performed reliably and met the design expectations. Each ping pong ball was successfully launched, with the trajectory consistently reaching a height of over 1.5 meters. This confirmed that the flywheel, feeder timing and orientation were well-calibrated. The successful firing validated the mechanical and electrical integration of the launching system.

Chapter 8: Evaluation and Areas for Improvement

8.1 Key Findings

8.1.1 Mechanical

The system underwent a thorough inspection to assess its stability and robustness. Upon activation, the ping pong balls were securely held in the hopper, demonstrating reliable storage. Additionally, the robot was firmly secured, with no loose components detected, ensuring safe operation.

8.1.2 Electrical

The wirings were inspected during the run, and it was kept tidy and secured. The wires from each component were bunched up and taped together using electrical tape for easy accountability and traceability, should troubleshooting need to be done. All of the wires were tucked nicely within the third level of the Turtlebot3.

8.1.3 Software

Initial testing showed nearly perfect navigation performance in the Gazebo simulation environment. However, transferring the same navigation code to the physical Turtlebot3 presented multiple challenges.

Firstly, intermittent connection issues caused delays or losses in ROS2 command messages, leading to unreliable movements and collisions with maze walls. These dropped commands also introduced significant noise in the Cartographer-generated maps, creating phantom obstacles and inaccuracies.

Secondly, our software made idealistic assumptions, particularly with the implementation of the pure pursuit navigation algorithm. While reverse movements were rare in simulation, the physical robot frequently navigated in reverse, impairing LIDAR performance due to our feeder mechanism placement. Additionally, our thermal sensor processing was based on hard-coded thresholds, which proved insufficient as actual target temperatures varied significantly between runs.

Lastly, extensive parameter tuning was required but underestimated. Parameters such as obstacle padding and minimum walkable thresholds needed precise adjustments. Combined with the inconsistent results due to connection issues and idealistic assumptions, achieving reliable performance through tuning proved challenging.

Time and manpower constraints further impacted our testing approach. In an ideal scenario, modular unit testing would verify each subsystem in isolation with well-defined stubs, followed by integration tests to evaluate component interactions. Comprehensive system testing in a representative maze environment would confirm overall performance against specified metrics.

8.2 Area for Improvement

A key area for improvement lies in optimising the robot's design to better align with its core mission objectives. As the primary goal is to detect and signal heat zones, rather than utilising the full capacity of nine balls, it is possible to reduce the ball-carrying requirement. This reduction would enable the creation of a shorter and more compact robot, enhancing its manoeuvrability in tight maze environments. A more compact design would also facilitate navigation through narrow passageways and ramps, thereby reducing the risk of getting stuck and improving overall performance. By prioritising mission-critical functionality over unnecessary payload, we can streamline the robot’s design for greater efficiency and reliability.

Conclusion

The TurtleBot3 Search and Rescue project successfully showcased an integrated approach to autonomous robotic navigation, thermal signal detection, and mechanical flare deployment in a simulated emergency setting. By merging hardware reliability with responsive control systems and sensor-driven decision-making, the robot effectively identified survivor zones through heat signatures and responded with visual signals using ping pong ball flares.

Throughout the project, the team tackled significant engineering challenges spanning mechanical design, electrical integration, and software architecture. The robot's modular design enabled systematic troubleshooting and iterative enhancements, while adherence to functional and performance requirements ensured a focus on core mission objectives.

This project not only demonstrated the potential of autonomous robotics in constrained environments but also underscored the importance of systems thinking and interdisciplinary collaboration. The insights gained provide a robust foundation for future advancements in robotics, signal communication, and iterative prototyping.

References

TurtleBot3 (robotis.com). (2024). Retrieved from https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/

franciscodias.net, X. site: (n.d.). Arduino Controlled Ping Pong Balls Launcher. Instructables. https://www.instructables.com/Pingo-a-Motion-Detecting-and-High-Accuracy-Ping-Po/

In-Depth: Interface L298N DC Motor Driver Module with Arduino. Retrieved from https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/

Open Robotics. *ROS 2 Documentation (Humble)*. Retrieved April 23, 2025, from https://docs.ros.org/en/humble/

Adafruit. *AMG8833 IR Thermal Camera Breakout*. Retrieved April 23, 2025, from https://learn.adafruit.com/adafruit-amg8833-8x8-thermal-camera-sensor

Links to Other Resources

Github Repository:
https://github.com/t-leongchuan/r2auto_nav_CDE2310_grp4

3D Printed files:
https://github.com/t-leongchuan/r2auto_nav_CDE2310_grp4/tree/main/docs/stl_files

User Documentations:
https://github.com/t-leongchuan/r2auto_nav_CDE2310_grp4/blob/main/docs/End%20User%20Documentation%20(2).pdf
https://github.com/t-leongchuan/r2auto_nav_CDE2310_grp4/blob/main/docs/Mechanical%20Instructions.pdf

Photos:
https://github.com/t-leongchuan/r2auto_nav_CDE2310_grp4/blob/main/docs/Final%20Turtlebot%20Pictures.pdf

Appendix A: G1 Documentation

Literature Review / Research

2.1 Autonomous Navigation in Unknown Environments / Software

Autonomous navigation has been extensively researched in the field of robotics. The Turtlebot, which serves as the platform for this mission, is equipped with LIDAR sensors and utilises the Robot Operating System (ROS) for mapping and path planning. Techniques such as Simultaneous Localisation and Mapping (SLAM) allow for real-time mapping and localisation, making them ideal for navigating unknown environments like the maze in this project.

2.2 Flare Subsystem

2.2.1 Firing Mechanisms

Various mechanisms for launching projectiles have been explored in robotics, including spring-loaded systems, flywheel launchers, and servo-driven catapults. These mechanisms have been widely applied in competitive robotics and previous iterations of CDE2310.

- Spring-loaded systems: deploy multiple flares using pre-compressed springs, lightweight and low-powered
    - Advantages: simple mechanism and lightweight
    - Disadvantages: reloading difficulties, limited range
- Flywheel Launchers: energy stored in the spinning flywheel is transferred to the object when the flywheel's rotational motion is suddenly stopped or redirected
    - Advantages: powerful launcher, high durability, quick reset
    - Disadvantages: power consumption
- Servo-Driven Catapults: operate on the same principle as traditional catapults but with servo motors providing controlled release
    - Advantages: simple mechanism
    - Disadvantages: limited launch force, reliability and reloading difficulties

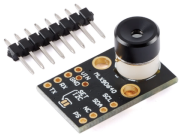
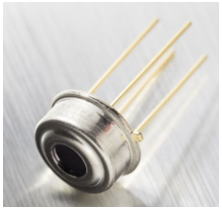
2.2.2 Carrying and Reloading

There are several design options for carrying and reloading the ping pong balls into the firing mechanism. We explored real-life examples such as gun magazines and revolvers, but ultimately, we decided on a simple ball-holding funnel to feed the balls into the system.

Additionally, ball-holding mechanisms like single-stacking and double-stacking configurations have been investigated primarily to optimise the design by minimising the height of the chute used to contain the balls, rather than influencing the accuracy of firing. Incorporating these

design considerations ensures the robot remains compact and efficient while fulfilling the flare deployment requirements.

2.3 Heat Detection

Type	Specifications	Advantages
MLX90640BAA (Thermal IR Array Sensor) [SGD 40-50] 	32x24 resolution thermal camera module	• Capable of measuring surface temperatures without physical contact. • Wide field of view (FOV) configurations: 55°x35° (standard) or higher for broader coverage. • Compact and energy-efficient
MLX90641BAB ($40-50) 	Higher resolution variant with a 16x12 sensor grid.	• Optimised for applications requiring mid-range thermal detection. • Offers faster refresh rates, improving detection accuracy in dynamic environments. • Compatible with Raspberry Pi for easy integration into robotics projects. • Lower power consumption compared to the MLX90640BAA, better suited for extended operations.
Infra-red thermometer ($10) 		• Much Cheaper • Single point detection with a narrow field of view • Possible to stitch together a thermal map using a few point temperatures • Attach onto servo to sweep an area.

2.4 Ramp Scaling

When scaling a ramp, key factors include traction, torque, and weight distribution. Traction is the friction between the tires and the surface, essential for grip and preventing slipping. Torque provides the rotational force needed to overcome gravity and move uphill. Weight distribution ensures balanced grip across all tires, especially on steeper inclines, helping maintain traction. Together, these factors determine the vehicle's ability to efficiently climb a ramp.

Concepts Design

3.1 Autonomous Navigation

When using LiDAR for mapping and obstacle detection in an autonomous system, the robot scans its surroundings and measures the time it takes for the beams to return after hitting a wall. This data allows the robot to detect walls or obstacles at varying distances, such as identifying a wall when it is within 10 cm. The distance data from the LiDAR sensor is processed by an algorithm that continuously scans and updates the robot's understanding of its environment. If a wall is detected within this threshold, the robot can respond by triggering a collision avoidance algorithm. This algorithm signals the robot to adjust its path to avoid the wall and continue navigating the environment.

The LiDAR sensor not only helps detect obstacles like walls but also contributes to creating a map of the environment. As the robot moves through the maze, the SLAM algorithm continuously integrates new LiDAR data, updating its position and refining the map in real-time. When the robot encounters an obstacle such as a wall, it updates the map with the new information, helping the robot avoid this area in future movements. The combination of localisation and mapping ensures that the robot can navigate around walls and other obstacles while maintaining an accurate and consistent map of its surroundings.

3.2 Heat Detection

For the heat detection, we will use MLX90640BAA Thermal IR Array Sensor, which is ideal for pinpoint detection of heat sources. A single sensor, positioned at the front of the TurtleBot3 to provide a wider detection range. These sensors will be faced forward to ensure the robot can track the heat signal in front of it, allowing the robot to follow the heat source effectively.

The algorithm driving the robot's behaviour will operate as follows: When a heat signal is detected, the heat sensor sends a signal to the Raspberry Pi to process the data. The TurtleBot3 will then navigate toward the heat signal by adjusting its movement based on the sensor readings. As the robot gets closer to the heat source, it will continuously check the distance. Once the sensors detect that the robot is 10 cm away from the heat signal, the algorithm will trigger the firing mechanism to launch the first ball, marking the completion of the first step in the mission.

3.3 Flares

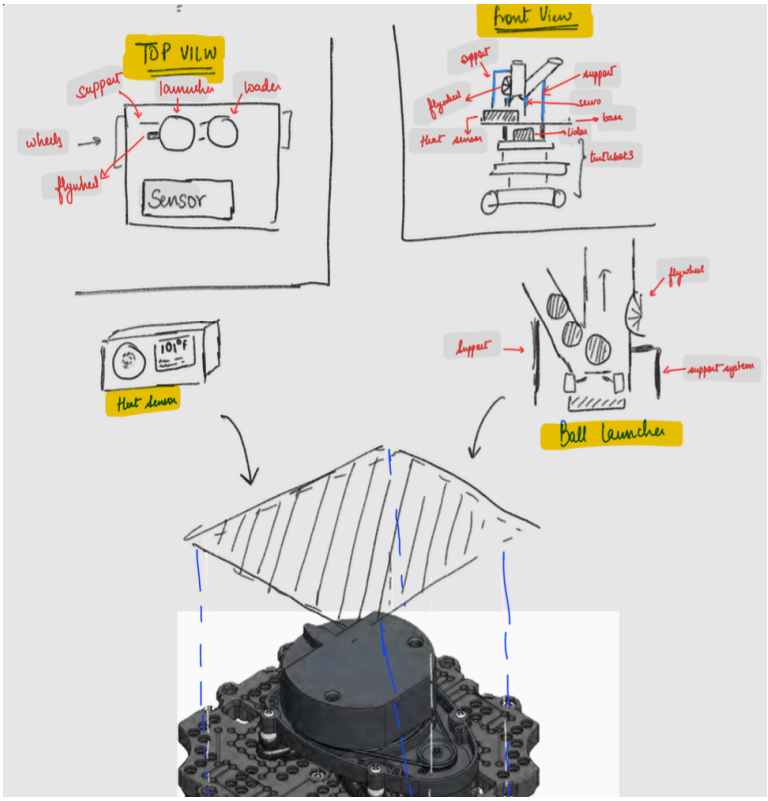
The flare launching mechanism operates using two servo motors and a flywheel system. When triggered, the first servo motor pushes a base upward, moving a ball toward the flywheels, which rapidly spin to launch the ball forward. This ensures a controlled and consistent launch with sufficient force to reach the target area.

After the ball is launched, the servo motor lowers the base, allowing the next ball to roll onto the plate from the storage mechanism. The system then waits for 2 seconds before repeating the cycle to ensure smooth operation and prevent jamming. This controlled timing allows the launcher to fire multiple flares efficiently while maintaining accuracy and reliability.

3.4 Ramp scaling

As the TurtleBot3 moves onto the ramp, it adjusts velocity and increases torque to counteract gravity. The motor controller detects resistance and boosts power to maintain forward motion, preventing stalling. Once at the top, the bot reduces torque to stabilise its speed, ensuring smooth and efficient ramp navigation.

Preliminary Design



This design sketch showcases a robotic system designed for search and rescue (S&R) operations, specifically for detecting heat signals and launching flares (ping pong balls) to indicate survivor locations.

Key Components and Their Functions:

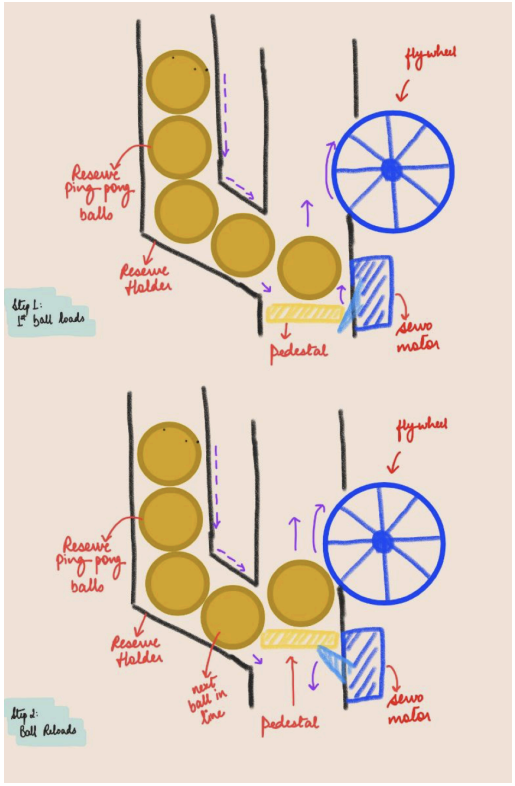
Structural Design:
The robot consists of a multi-layered chassis with an additional upper layer housing critical components like the sensor and launching mechanism. The modular design suggests easy assembly and adaptability.

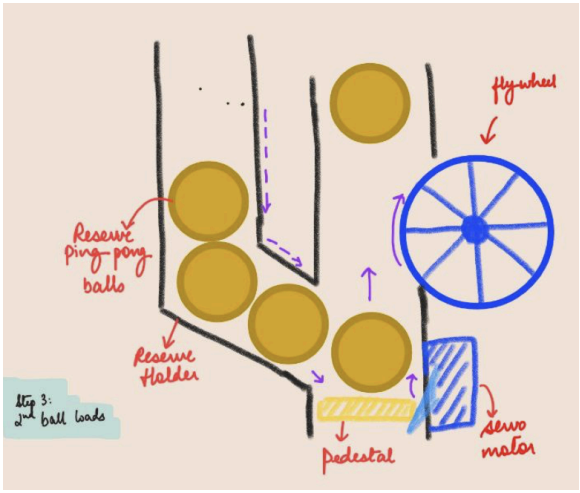
Sensor System:
- The heat sensor is likely mounted at the top to effectively scan the environment.
- LiDAR Module: Positioned at the top, the 2D LiDAR helps with mapping the maze and avoiding obstacles. Given the constraints of the challenge (no line-following navigation), LiDAR is essential for autonomous movement.

Ball Launching Mechanism:
- The ball launching mechanism is added to the top of the robot by adding an additional layer and adding supports where needed.
- Flywheels & Support System: The ball launcher appears to rely on a flywheel mechanism, where ping pong balls are fed into the system and launched using high-speed spinning wheels.
- Loader & Feeder: A separate loader system ensures continuous feeding of balls into the launcher, likely controlled by servos. The balls are stored in a hopper before being fired.

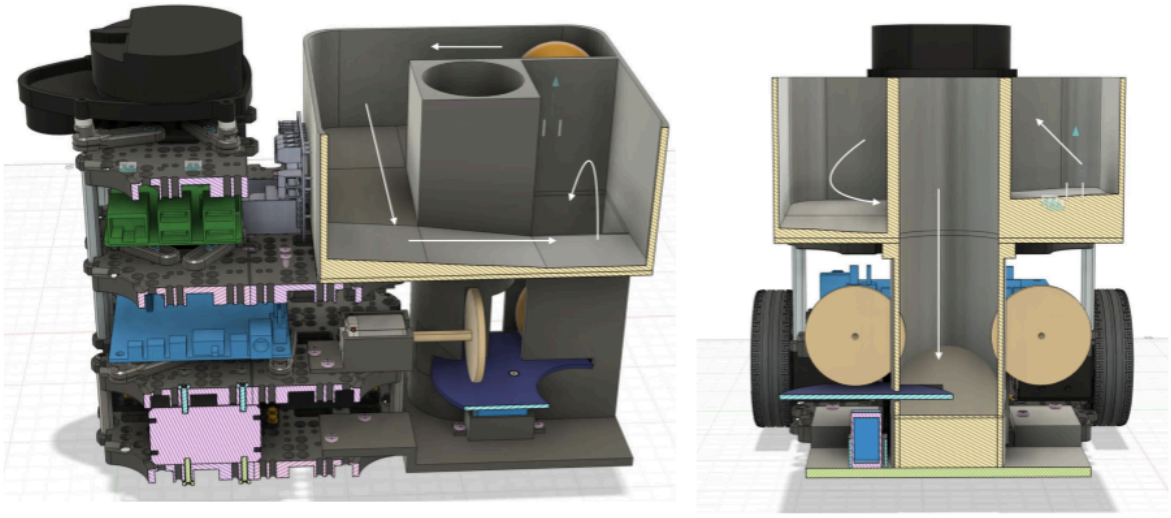
Mechanical Considerations
- The additional supports ensure that all components remain stable during movement and firing.
- Given the need for precise targeting, the launcher's positioning relative to the LiDAR and heat sensor is crucial.
- The robot's weight distribution is carefully managed to prevent imbalance due to the firing force.

Firing Mechanism[2]





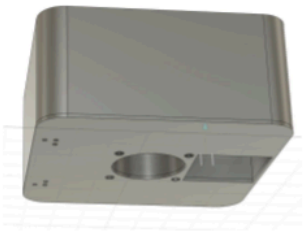
Appendix B: PDR



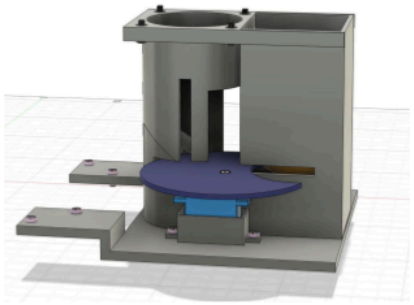
Machine Elements

Item	Qty	Additional Note	Material
AMG8833 Mount	1	-	PETG
Motor Mount	2	-	PETG
Flywheels	1	-	PETG
Rotating Feeder	1	-	PETG
PH_M3x8mm_K	10	Screw	Stainless Steel
PH_M2x8mm_K	6	Screw	Stainless Steel
PH_M3x6mm_K	12	Screw	Stainless Steel
NUT_M2	6	Nut	Stainless Steel
NUT_M3	6	Nut	Stainless Steel

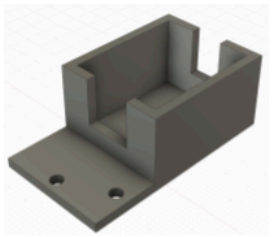
Assembly Plan

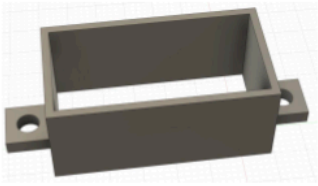


Storage Unit



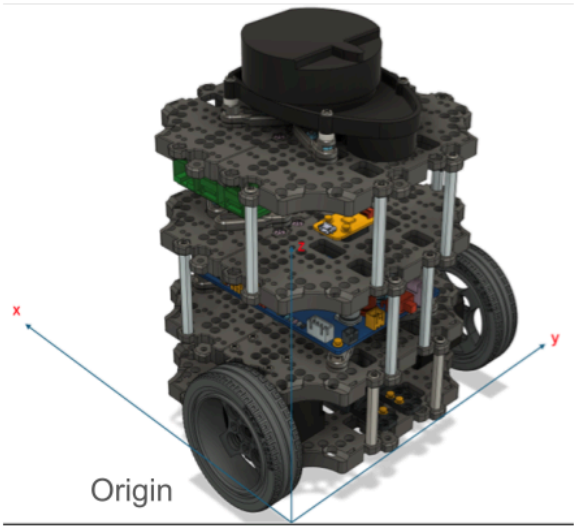
Firing and Feeder Mechanism





Motor and Servo Mounts

Centre of Mass - Origin

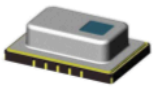


Power Budget

Components	Voltage (V)	Current (A)	Qty	Power (W)	Remarks
During Boot Up					
Initial Boot Up	11.09	0.634	1	7.03	As the buzzer will sound
				7.03	
Standby					
During Standby/Idle	11.09	0.523	1	5.86	-
L298 Motor Driver*	2	2	1	4	66% duty cycle
Servo Motor	4.99	0.153	1	0.763	Maximum duty cycle
12V DC Motor*	11.1	0.26	2	5.88	
				16.5	
Operations					
During Operation, rteleop	11.09	0.733	1	8.14	Measured during maximum translational velocity = 0.22 m/s
AMG8833 Infrared Array Sensor*	3.3	0.0045	1	0.01485	Operating at 10 fps
				8.155	

*~~Taken from the respective datasheet~~

Selection of Sensors and Actuators



AMG8833

Key Specifications:
Sensor Array: 8x8 (64 pixels)
Temperature Range:
-20°C to 100°C
Supply Voltage: 3.3V to 5V

Thermal Imaging, compact, cost-effective

FS90R

Key Specifications:
Control Element: Speed Control
Rotate continuously (360°) in either direction

Suitable for both loading and limiting the ping pong balls.

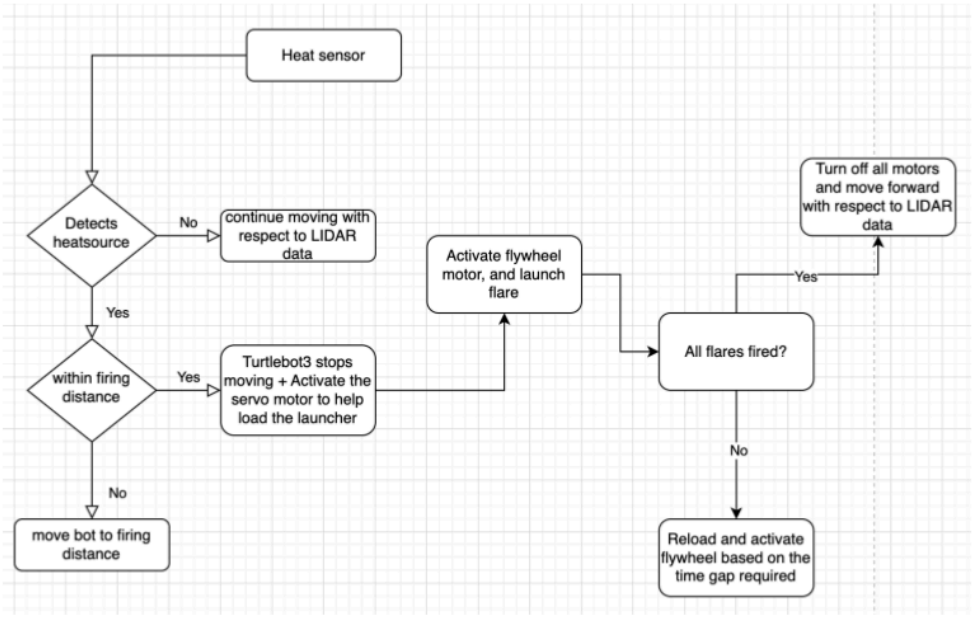
L298N

Key Specifications:
Control the speed of the 6V DC motor with PWM signals
Handle from 5V to 35V and provide up to 2A of continuous current per channel.

Voltage and Current Control. Motor speed control

RS-385

Key Specifications:
Operating Voltage: 6V to 12V DC
No-load Speed: 3000 ± 500 RPM
Loaded Speed: 4000 ±1000 RPM
Starting Torque: 40 g*cm
Weight: 17.5 grams


Heat sensor
Detects heatsource
No
continue moving with respect to LIDAR data
Yes
within firing distance
Yes
Turtlebot3 stops moving + Activate the servo motor to help load the launcher
No
move bot to firing distance
Activate flywheel motor, and launch flare
All flares fired?
Yes
Turn off all motors and move forward with respect to LIDAR data
No
Reload and activate flywheel based on the time gap required

Procurement Plan

Item	Quantity	Additional Note	Procurement Method	Price / Unit
Firing and Feeding Funnel	1		3D Print	14.11
Storage Unit	1		3D Print	
Electronic Mounts	3		3D Print	
Rotating Feeder	1		3D Print	
Fasteners	40		E2A Lab	
AMG8833	1	Heat Sensor	E2A Lab / Buy	27.61
Continuous Rotation Micro Servo - FS90R	1	Servo Motor	Buy	10.94
DC Motor - RS-385	2	Flywheel Motor	Buy	10.62
L298N	1	Motor Driver	E2A Lab	

Total Price = **$53.28**
(exclusive of delivery fee)

Timeline

Time	Plan	Key Achievements & Test	Technical Performance Measure (TPM) (things to test)	Measure of effectiveness (MOE)	Subsystem
Week 8	Hardware Fabrication & ROS2 Setup	Fabricate mechanical parts, test electrical components. Implement ROS2 skeleton, verify node communication.	Components within tolerance specs. ROS2 nodes launch & interact correctly.	All fabricated parts meet design requirements. ROS2 builds & runs without errors.	Mechanical, Electrical, Software
Week 9	Autonomous Navigation & FSM Development Payload Assembly	Test Cartographer SLAM for mapping. Implement & refine FSM transitions. Test Thermal Sensor for heat detection.	Remote Computer able to generate a graph from given Occupancy Grid. Heat source detected within 2m ± 0.5m. Firing delay < 1s.	TurtleBot correctly navigates without excessive looping. No excessive retries or false detections.	Software, Electronics, Mechanical
Week 10	Target Alignment & Firing Mechanism	Integrate Alignment Node for precise positioning. Test Thermal Sensor for heat detection. Implement & test Firing Mechanism.	Alignment accuracy ±3°.	TurtleBot aligns before firing.	Software, Electrical
Week 11	Full Mission Flow Testing & Debugging	Run full mission trials (Exploration → Alignment → Firing → Recovery). Optimize navigation efficiency & FSM transitions.	Full mission success rate > 90%. Pathfinding no detours.	Robot completes the mission without failures. FSM recovers from alignment or firing failures.	All

Appendix C: Datasheet of Electrical Components

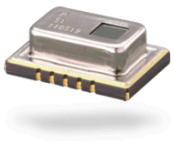
AMG8833 Infrared Camera

Panasonic
INDUSTRY

Infrared Array Sensor Grid-EYE

Surface Mount Type

AMG88xx（High performance type）



High precision infrared array sensor based on advanced MEMS technology

Feature

- Temperature detection of two-dimensional area: 8 × 8 (64 pixels)
- Digital output (capability of temperature value output)
- Compact SMD package (adaptively to reflow mounting)
- RoHS compliant

Types

Product name	Number of pixel	Operating voltage	Amplification factor	Part number	Tape and reel package (pcs)
Infrared array sensor Grid-EYE	64 (Vertical 8 × Horizontal 8 Matrix)	3.3 V	High gain	AMG8833	1000
			Low gain	AMG8834	
		5.0 V	High gain	AMG8853	
			Low gain	AMG8854	

Rating

Item	Performance	
	High gain	Low gain
Applied voltage	3.3 V ± 0.3 V or 5.0 V ± 0.5 V	
Temperature range of measuring object	0 °C to 80 °C +32 °F to +176 °F	−20 °C ～ 100 °C −4 °F to +212 °F
Operating temperature range	0 °C to 80 °C +32 °F to +176 °F	−20 °C ～ 80 °C −4 °F to +176 °F
Storage temperature range	−20 °C to 80 °C −4 °F to +176 °F	−20 °C ～ 80 °C −4 °F to +176 °F

Absolute maximum ratings

Item	Absolute maximum ratings	Terminal
Applied voltage	−0.3 V to 6.5 V	VDD
Input voltage	−0.3 V to VDD +0.3 V	SCL, SDA, AD_SELECT
Output sink current	−10 mA to 10 mA	INT, SDA
Static electricity (Human Body Model)	1 kV	All terminals
Static electricity (Machine Model)	200 V	All terminals

Characteristics

Item	Performance	
	High gain	Low gain
Temperature accuracy	Typ. ± 2.5 °C ±4.5 °F	Typ. ± 3.0 °C ±5.4 °F
NETD *1	Typ. 0.05 K (in 1 fps setting *2) Typ. 0.16 K (in 10 fps setting)	
Viewing angle	Typ. 60 °	
Current consumption	Typ. 4.5 mA（normal mode） Typ. 0.2 mA（sleep mode）	
Setup time	Typ. 50 ms（Time to enable communication after setup） Typ. 15 s（Time to stabilize output after setup）	

*1: It is calculated from 4 pixels of centers.
*2: fps: frame per second

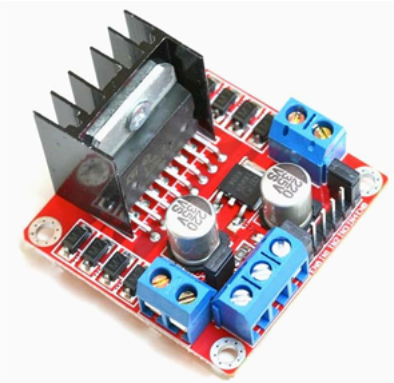
Performance

Item	Performance
Number of pixel	64 (Vertical 8 × Horizontal 8 Matrix)
External interface	I^2C
Frame rate	Typ. 1 fps or Typ. 10 fps
Operating mode *3	Normal Sleep
Output mode	Temperature output
Calculate mode	No moving average or Twice moving average
Temperature output resolution	0.25 °C 0.45 °F
Number of sensor address	2（I^2C slave address))
Thermistor output temperature range	−20 °C to 80 °C −4 °F to +176 °F
Thermistor output resolution	0.0625 °C 0.1125 °F

*3: Normal Mode : normal operation mode; Sleep Mode: detection is off (output and data reading not possible)

Refer to the following link for AMG8833 datasheet: AMG88xx

L298N Motor Driver



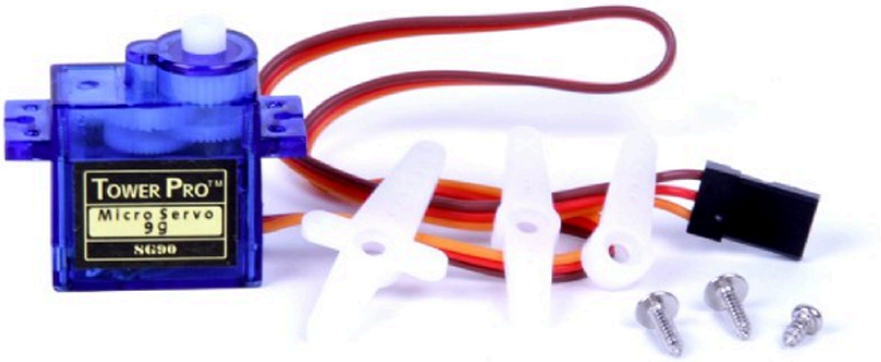
SKU: MDU-1049

Brief Data:

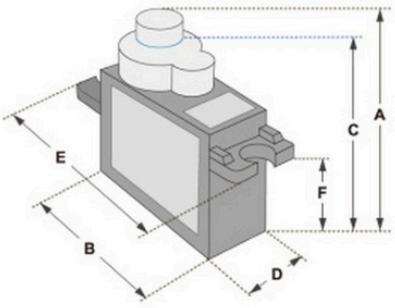
- Input Voltage: 3.2V~40Vdc.
- Driver: L298N Dual H Bridge DC Motor Driver
- Power Supply: DC 5 V - 35 V
- Peak current: 2 Amp
- Operating current range: 0 ~ 36mA
- Control signal input voltage range :
- Low: -0.3V ⩽ Vin ⩽ 1.5V.
- High: 2.3V ⩽ Vin ⩽ Vss.
- Enable signal input voltage range :
    - Low: -0.3 ⩽ Vin ⩽ 1.5V (control signal is invalid).
    - High: 2.3V ⩽ Vin ⩽ Vss (control signal active).
- Maximum power consumption: 20W (when the temperature T = 75 ℃).
- Storage temperature: -25 ℃ ~ +130 ℃.
- On-board +5V regulated Output supply (supply to controller board i.e. Arduino).
- Size: 3.4cm x 4.3cm x 2.7cm

Refer to the following link for L298N datasheet: L298N Motor Driver.pdf

SG90 180° Positional Servo Motor



Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.



Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Refer to the following link for SG90 datasheet: SERVO MOTOR SG90 DATA SHEET

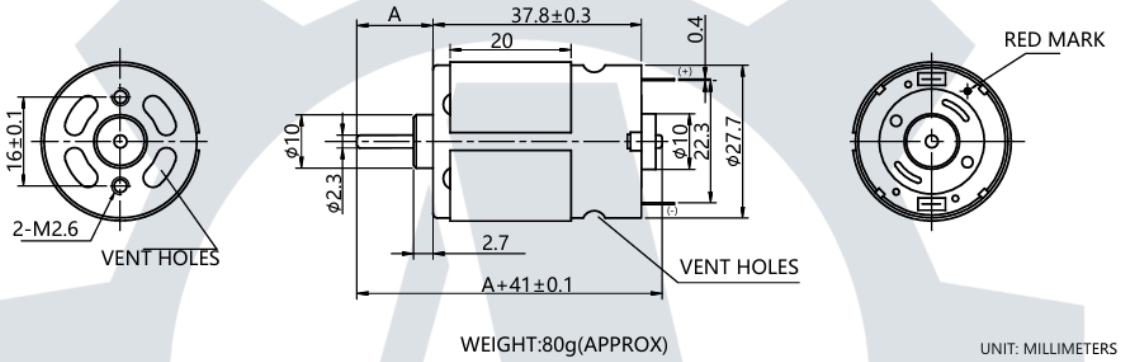
RS385 12V DC Motor

FARS-385

(voltage, current, torque, etc free customization)

典型应用 Typical Applications:		轴长 Shaft Length A(mm)
真空吸尘器、激光打印机、电动螺丝刀	Vacuum cleaner; Laser printer; Electric screwdriver	Customizable

外形尺寸 DIMENSION

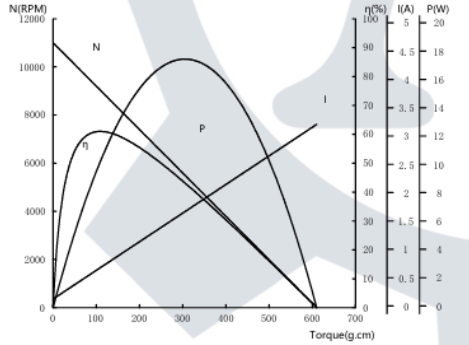


WEIGHT:80g(APPROX)

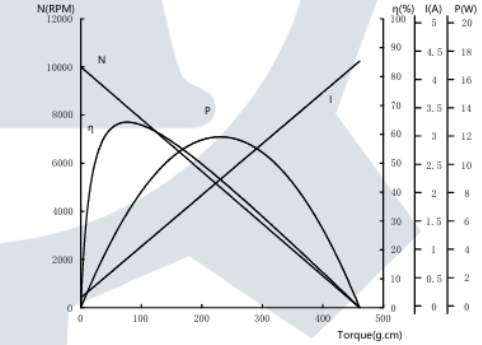
UNIT: MILLIMETERS

Micro brush DC electric motor performance curve

FARS-385SH-17110 24V



FARS-385SH-2267 12V



微型有刷直流电机性能参数表 Micro brush DC electric motor data table

型号 Model	电压 Voltage		空载 No Load		额定负载 RATED LOAD						堵转 Stall		
	适用范围 Operating range	额定 Rated	转速 Speed	电流 Current	转速 Speed	电流 Current	扭矩 Torque		输出功率 Output	效率 Efficiency	扭矩 Torque		电流 Current
	VDC		RPM	A	RPM	A	mN.m	gf.cm	W	%	mN.m	gf.cm	A
FARS-385SH-17110	18-28	24	11000	0.15	9034	0.69	10.69	109	10.11	61.0	59.80	610	3.16
FARS-385SH-2267	6.0-14.0	12	10000	0.17	8340	0.85	7.50	76.5	6.55	64.2	45.00	459	4.29

Refer to the following link for RS385 datasheet: [FARS-385 RS-385 RC-385 OD 28mm micro dc electric motor specs]